Elsevier Editorial System(tm) for Computers & Education
Manuscript Draft

Manuscript Number: CAE-D-11-00799R2

Title: The Effect of Positive Feedback in a Constraint-Based Intelligent Tutoring System

Article Type: Research Paper

Keywords: intelligent tutoring systems; pedagogical issues; teaching/learning strategies

Corresponding Author: Professor Antonija Mitrovic, Ph.D

Corresponding Author's Institution: University of Canterbury

First Author: Antonija Mitrovic, Ph.D

Order of Authors: Antonija Mitrovic, Ph.D; Stellan Ohlsson, Professor; Devon Barrow

Abstract: Tutoring technologies for supporting learning from errors via negative feedback are highly developed and have proven their worth in empirical evaluations. However, observations of empirical tutoring dialogues highlight the importance of positive feedback in the practice of expert tutoring. We hypothesize that positive feedback works by reducing student uncertainty about tentative but correct problem solving steps. Positive feedback should communicate three pieces of explanatory information: (a) those features of the situation that made the action the correct one, both in general terms and with reference to the specifics of the problem state; (b) the description of the action at a conceptual level and (c) the important aspect of the change in the problem state brought about by the action. We describe how a positive feedback capability was implemented in a mature, constraint-based tutoring system, SQL-Tutor, which teaches by helping students learn from their errors. Empirical evaluation shows that students who were interacting with the augmented version of SQL-Tutor learned at twice the speed as the students who interacted with the standard, error feedback only, version. We compare our approach with some alternative techniques to provide positive feedback in intelligent tutoring systems.

Response to Reviewers - NOTHING identifying the author should be listed here

We have made all the changes requested by Reviewer 3 (there was only one review available in the second round).

Title Page -- anything identifying the author should be on this page.

The Effect of Positive Feedback in a Constraint-Based Intelligent Tutoring System

Antonija Mitrovic[1], Stellan Ohlsson[2], and Devon K. Barrow[1]

[1] Department of Computer Science and Software Engineering, University of Canterbury, Private Bag 4800, Christchurch 8140, New Zealand
tanja.mitrovic@canterbury.nz

[2] University of Illinois at Chicago, Department of Psychology, 1007 West Harrison Street, Chicago, IL 60607, USA
stellan@uic.edu

Corresponding author:

Antonija Mitrovic,
Private Bag 4800
Christchurch 8140
New Zealand
Email: tanja.mitrovic@canterbury.nz
Fax: 64-3-364-2569
Voice mail: 64-3-364-2987, ext. 7757

The Effect of Positive Feedback in a Constraint-Based Intelligent Tutoring System

Abstract

Tutoring technologies for supporting learning from errors via negative feedback are highly developed and have proven their worth in empirical evaluations. However, observations of empirical tutoring dialogues highlight the importance of positive feedback in the practice of expert tutoring. We hypothesize that positive feedback works by reducing student uncertainty about tentative but correct problem solving steps. Positive feedback should communicate three pieces of explanatory information: (a) those features of the situation that made the action the correct one, both in general terms and with reference to the specifics of the problem state; (b) the description of the action at a conceptual level and (c) the important aspect of the change in the problem state brought about by the action. We describe how a positive feedback capability was implemented in a mature, constraint-based tutoring system, SQL-Tutor, which teaches by helping students learn from their errors. Empirical evaluation shows that students who were interacting with the augmented version of SQL-Tutor learned at twice the speed as the students who interacted with the standard, error feedback only, version. We compare our approach with some alternative techniques to provide positive feedback in intelligent tutoring systems.

Keywords: intelligent tutoring systems, pedagogical issues, teaching/learning strategies

1. Introduction

Interactive learning environments that support the acquisition of cognitive skills must provide opportunities for the learner to practice the target skill. In the course of practice, the learner's representation of the skill-to-be-learned undergoes distinct types of changes. In a recent review of computational models of skill acquisition (Ohlsson, 2008, 2011), we distinguished nine different *modes of learning*, each of which has been implemented in running simulation models and tested against data from human learners: proceduralization of direct instructions, analogical inference from a prior problem for which the solution is known, reasoning from prior declarative knowledge about the task, generalization from solved examples, caching of correct or useful outcomes in response to positive feedback, error correction in response to negative feedback, strategy shifts, the elimination of redundancies and optimization by adaptation to statistical regularities in the task environment. Learning requires multiple cognitive mechanisms because each mechanism capitalizes on a different source of information that might be available to a learner during skill acquisition. For example, to generalize from a solved example requires a different cognitive process than to optimize a strategy by identifying redundant steps in a memory trace. Although the level of empirical support for these learning mechanisms varies from mechanism to mechanism, their psychological reality is not in doubt. In conjunction, the nine modes of change constitute a computational theory of how cognitive skills improve during practice.

The multiple mechanisms theory of learning suggests that instruction works by providing the different learning mechanisms with the information they need as input. For example, an instructor might support learning by generalizing from a solved example by providing such an example but also by highlighting its general features. Alternatively, an instructor can help a learner by reminding him or her of an analogical problem; by pointing out a useful shortcut; and so on. Each learning mechanism requires particular types of information, and instruction can support learning by providing more information of the relevant types than would otherwise be available to the learner. This *information specificity* principle implies that there are nine modes of instruction, corresponding to the nine modes of learning.

This view has consequences for the design of Intelligent Tutoring Systems (ITS), educational software systems that use artificial intelligence techniques to adapt the instruction to the individual student. In the early years of intelligent tutoring system research, system designers focused almost exclusively on supporting error correction by providing negative feedback (e.g., Brown & Burton, 1978; Sleeman, Kelly, Martinak, Ward & Moore, 1989; Spohrer, Soloway & Pope, 1985). The main tutoring moves supported by many early systems were to call attention to errors and to provide explanations or other types of information that might help a student unlearn an error. This focus was rooted less in learning theory than in common sense: When a student is solving his or her practice problems correctly, there seems to be little reason to interrupt his or her efforts to provide instruction, so tutoring systems were designed to intervene when they spotted incorrect steps or solutions.

In past work, we developed a technology called *constraint-based modeling* (CBM) for supporting learning from errors (Mitrovic & Ohlsson, 2006; Ohlsson, 1992; Ohlsson & Mitrovic, 2007). This technology rests on the idea of expressing the domain knowledge in terms of constraints, and then using the constraints to detect student errors and to decide what information to include in a tutoring message. The alternative model-tracing technology rests on the idea of expressing domain knowledge as a set of correct and buggy rules and then matching rules against student actions in order to decide what content to include in a tutoring message. There are multiple tutoring systems built with each technology that react to student errors with adapted feedback. Empirical evaluations have shown beyond doubt that those tutoring systems help students learn (Anderson et al., 1995; Koedinger et al., 1997; Mitrovic, 2006; Mitrovic, Suraweera, Martin & Weerasinghe, 2004).

However, the multiple-modes perspective indicates that for ITSs to provide maximal support for learning, they should support all modes of learning. Although tutoring systems have been developed that teach in other ways than by reacting to student errors, the technologies for supporting other modes of learning have not yet achieved the same level of maturity as the various teach-to-error technologies. To increase the pedagogical power of tutoring systems will require the development of mature technologies for supporting all modes of learning.

The purpose of this paper is to report the effect of adding a positive feedback capability to a well-established, constraint-based tutoring system, SQL-Tutor (Mitrovic & Ohlsson, 1999), which in its standard form teaches primarily by reacting to students' errors. We start Section 2 by discussing positive feedback provided by human teachers, and then present our Uncertainty Reduction hypothesis: positive feedback is effective because it reduces uncertainty in the student's knowledge. The contribution of this paper is in the proposal of how positive feedback can be provided, focusing on identifying situations when positive feedback is useful, as well as the content of positive feedback. Section 3 describes the implementation of the positive feedback facility within SQL-Tutor. We conducted an evaluation study which is discussed in Section 4. The final section presents the conclusions and also some alternative techniques to provide positive feedback in intelligent tutoring systems.

2. The Problem of Positive Feedback

There is widespread belief that positive feedback supports learning. Consistent with this, positive feedback appears in dialogues between experienced human tutors and students. Table 1 shows four excerpts from dialogues between tutors and students in the domain of Java programming (Boyer et al., 2008). In Episode 1, the student contributes a problem solving step and the tutor confirms its correctness. In Episode 2, the tutor provides guidance as to the next subgoal to be achieved. The student takes two steps, expressing uncertainty each time, and receives confirmation that the steps are indeed correct. In Episode 3, the tutor initiates the interaction by raising a problem with the student's solution and prompting the student to repair it. The student suggests a repair but signals uncertainty by stating it as a question; the tutor verifies that the suggestion is correct. This cycle then repeats. In Episode 4, the same suggestion-verification sequence occurs once

again, but this time the tutor also opts to supply subject matter content intended to help the student understand why his or her suggestion was correct.

Place Table 1 here

Excerpts from tutoring dialogues provide existence proof but do not allow us to infer prevalence. How frequently does positive feedback occur in tutorial dialogues, both in terms of absolute numbers and in relation to other types of tutoring moves? Cade, Copeland, Person and D'Mello (2008) analyzed 40 recordings of one-hour tutoring sessions. The tutors were eight expert mathematics and science tutors. They tutored 30 different students. The interactions were analyzed in terms of eight mutually exclusive tutoring modes. The mode called "Scaffolding" was defined as "the tutor intervenes when necessary so that the student can successfully arrive at a correct solution", which we assume includes immediate, in-context feedback. The results showed that 30% of the tutoring moves fell in this category, and that this was the most frequent of the eight modes. However, the presentation of the results did not differentiate between negative and positive feedback.

Boyer et al. (2008) analyzed a corpus of 43 one-hour tutoring dialogues between students from a university-level introductory Java computer programming class and fourteen tutors of varying experience. The students created Java statements, and the tutors communicated with students from a separate room by typing their feedback onto the students' screens. Automatic coding of the interaction logs identified 1,243 tutor moves that occurred immediately after an event in which the student typed a piece of code that was later deleted or replaced. In this corpus, 184 out of the 1,243 feedback messages were classified as providing positive feedback indicating a correct move on the part of the students. In addition, there were 89 cases of content-free praise, but only 24 negative feedback messages indicating a student error. All three types of feedback were less frequent than tutorial statements that provided subject matter information without explicitly labeling a student move as correct or incorrect; there were 828 of the latter. In short, informative positive feedback messages were more common than either content-free praise or negative feedback messages, but they nevertheless constituted only 15% of all tutoring moves.

The relatively low frequency of positive feedback messages in these studies raises the crucial and more difficult question whether positive feedback contributes to the effectiveness of learning. Ohlsson et al. (2007) and Di Eugenio, Fossati, Ohlsson and Cosejo (2009) coded a corpus of 54 tutoring dialogues, nearly 34 hours of interaction, in the field of computer science, specifically elementary data structures like linked lists. The tutors were two computer science professors, one of whom had extensive tutoring experience and one of whom was a junior faculty member without extensive tutoring experience. The impact of pre-test performance, the duration of the tutorial interaction, the identity of the tutor, number of positive feedback messages and number of negative feedback messages were assessed through a multiple regression with the pre-to-post-test gain as the dependent variable. The regression analysis revealed a strong effect of both

pre-test score and time on task on learning gains, as one would expect. However, there was no relation between tutor and gain scores, indicating that the large gap in experience between the two tutors did not impact pedagogical effectiveness. The central result was a robust effect of the number of positive feedback messages on gain scores. A regression model with pre-test score, time on task, number of positive feedback messages and number of negative feedback messages as predictors accounted for 32% of the variance in the students' gain scores. Of the four predictors, only pre-test score and number of positive feedback messages were significantly related to gain scores. Pre-test scores correlated negatively with gain scores. The results strongly support the conclusion that positive feedback is instrumental in raising gain scores.

In short, available data support the view that human tutors deliver positive feedback, that they deliver positive feedback more often than negative feedback, that the absolute frequency of positive feedback is low but that positive feedback is nevertheless instrumental in producing the high gain scores obtained via tutoring.

2.1. Why is Positive Feedback Effective?

How does positive feedback support learning? That is, if a student produces a correct problem solving step or a correct solution, why does it help him or her to be told, "that was the right move" or "yes, that's the correct solution"? In order to elicit positive feedback from his or her learning environment, the learner has to produce an appropriate or correct step or solution. The situation is illustrated in Table 1: The students produce correct steps or solutions, and the tutor confirms them. But if the learner already knows enough to take the right problem solving step or propose a correct solution, then it would seem that he or she has already mastered the relevant part of the skill-to-be-learned. There is thus nothing for him or her to learn with respect to that particular part of the subject matter, and the tutorial intervention serves no purpose. If so, why is positive feedback related to learning gains?

One possible resolution to this paradox is that students' problem solving steps are often tentative. At the outset of learning, a student does not yet know what to do, so he or she must act on the basis of general methods and heuristics. In the domain of consumer electronics, an example of a general heuristic is *if you don't know how to use the device, push any button and observe what happens*. In algebra, a student might operate with the heuristic, *if you can't decide which transformation is the right one, do any transformation you can think of and see what happens*. Heuristics and dispositions of this weak, general sort do not guarantee that the actions they recommend are correct or useful, but what little guidance they provide might be all the guidance a novice has. In short, novices do not yet know what they are doing, so they frequently guess.

Many tentative steps will of course turn out to be inappropriate or incorrect. However, other steps will turn out to be correct. Examples 2, 3 and 4 in Table 1 illustrate this type of situation. It is plausible that the main function of positive feedback is to reduce uncertainty associated with the latter category of steps. By confirming the correctness of a tentative step, the tutor can help the student encode and reproduce that step in future situations in which it is relevant. The consequence is that the correct step is taken with higher probability, less processing or both.

To transform this *Uncertainty Reduction Hypothesis* about learning into a principle for instruction requires answers to two questions: First, when, at which points during tutoring, is positive feedback beneficial? The second problem is how a tutoring system can compute the content of the positive feedback that a student should receive in any one situation.

2.2. When Should Positive Feedback Be Given?

A tutoring system could deliver positive feedback after every correct step. This is a computationally cheap solution, but it is unlikely to be productive. If the learner knows what to do, and knows that he or she knows what to do, then telling him or her that the step just taken was correct is unlikely to be useful. In addition, students who know what to do in most situations will see a large number of positive feedback messages and might learn to ignore them. The uncertainty reduction hypothesis claims that the pedagogical power of positive feedback is selective: Positive feedback is useful when (and perhaps only when) the student is uncertain but nevertheless happens to make the right move.

To implement this principle, an intelligent tutoring system must have some way to recognize that a student is uncertain. There are several possibilities. The system might collect response times and use them to identify problem states in which the student took longer than usual to decide what to do. The system might ask the student for subjective confidence ratings. The range of possibilities depends in part on the representations and the capabilities of the tutoring system itself. For example, a system with a video camera and facial expression recognition software might be able to detect facial signals of uncertainty like furrowed brows, but most systems will not have such sophisticated capabilities. A different approach is to collect statistical information about student behavior such as previous performances on particular practice problems, number of correct applications of a particular knowledge element, and so on. The question is how to use such information to accurately identify problem states in which a student hesitates and hence might benefit from positive feedback. After discussing what information should be included in a positive feedback message, we describe how this problem was solved in a constraint-based tutoring system.

2.3. What Information Should Be Conveyed?

Given that the current problem state has been identified as a plausible target for positive feedback, what information should the tutoring system deliver? What is the content of a helpful positive feedback message? If the main function of positive feedback is to reduce student uncertainty about the correctness of particular problem solving steps, then bare bones, content-free affirmative messages like “that’s right” or “great job” might be sufficient. The learning triggered by content-free affirmations might merely be to increase the strength of the associated knowledge structures, thereby reducing the uncertainty about their future application. Chi and co-workers have found significant effects of content-free instructional prompts (Chi et al., 2001) in the context of self-explanation.

However, it is likely that more elaborate and content-full feedback messages can provide additional benefit. If the student is truly guessing and has little or no sense of

why the step taken was appropriate, correct or useful, he or she might benefit from being told which features of the situation indicate that the step taken would turn out to be productive. For example, a student who after hesitation responds to an algebra problem like "5X + 15 = 3X – 5 + 2" by subtracting 3X from each side might benefit from being told, not only "that's right", but also that "whenever there are two terms with the same unknown on both sides of the equal sign, and the problem asks for the value of the unknown, subtracting one of those terms from both sides is a good move." The key point is to emphasize exactly which aspects of the problem indicate that this is a productive move (two terms with the same unknown, the terms distributed on both sides of the equal sign, the question asking for the value of the unknown). Acquiring a skill is to a large extent to learn when, under which circumstances, to do what. An elaboration of this sort on the part of a tutor can be seen in Example 4 in Table 1.

It is also plausible that a student who hesitates might have a weak concept about the meaning of the action he or she just performed. To continue the example, the student might not conceptualize the action "to subtract 3X from both sides of the equation" as "to combine the terms with same unknowns." He or she may have performed the action based on a shallow memory of what a teacher demonstrated, or a solved but incompletely understood textbook example. The level of abstraction is an important issue here: "to subtract 3X from both sides of the equation" is obviously too specific a conception, so a tutoring message might help by prompting the higher abstraction level, "to combine terms with the same unknown." In short, extending the feedback message by explicitly describing the action in the relevant way might help the student conceptualize the action in a useful manner.

Another reason why a student might benefit from being told how to conceptualize the action is that it helps him or her relate the action to the relevant goal or subgoal. An action might have multiple consequences, but only some of them are what made the action correct in the situation in which it was executed. In the example, the tutor might say "so now you have only a single term with X in it." The important consequence is that the number of terms *with the unknown* has been reduced, not that the total number of terms has been reduced, that the unknown occurs only the left of the equal sign, nor that the numerical value of the coefficient is lower. In general, there are different ways to conceptualize the consequences of a problem solving step, and a positive feedback message might be useful if it steers the student towards the most fruitful concept. We refer to this as the *micro-engineering* of the feedback messages. Several studies support the hypothesis that micro-engineering the content of feedback messages can influence learning gains (e.g., Zakharov, Mitrovic & Ohlsson, 2005).

To summarize, we propose that positive feedback should be delivered at those moments when the student is uncertain about what to do but nevertheless does the right thing (the *Uncertainty Reduction Hypothesis*). We further propose that a positive feedback message should communicate three pieces of explanatory information, over and above the bare bones confirmation that the step taken was correct: (a) those features of the situation that made the action the correct one, both in general terms ("multiple terms with the same unknown") and with reference to the specifics of the problem state ("you have X over here and X over there as well"); (b) the conceptually relevant description of the action at a conceptual level ("combine the terms with the unknown by subtracting 3X

from both sides of the equation"); and (c) the important aspect of the change in the problem state brought about by the action ("now you only have a single X-term").

3. Positive Feedback in a Constraint-Based System

We applied the design principles developed in the previous section by adding a positive feedback capability to an existing intelligent tutoring system called SQL-Tutor. The system already had the capability of providing negative feedback to support learning from error and it has been shown to be effective in multiple evaluation studies (Mitrovic & Ohlsson, 1999; Mitrovic, 2003; Mitrovic, Martin & Mayo, 2002). We summarize the domain and the system, how the positive feedback capability was implemented and finally the outcome of an empirical evaluation.

3.1. Constraint-Based Modeling and SQL-Tutor

SQL-Tutor is based on *constraint-based modeling* (CBM), a design philosophy for helping students learn from their errors (Mitrovic & Ohlsson, 2006; Ohlsson, 1992; Ohlsson & Mitrovic, 2007). The basic idea behind the constraint-based approach is that the target subject matter is encoded as a collection of knowledge elements called *constraints*. For example, *if you are driving in New Zealand, you should be driving on the left side of the road* is a constraint on traffic; *if you are adding two fractions by adding their numerators, those fractions must have the same denominator* is a constraint from arithmetic. In a constraint-based tutoring system, the student's solution is compared to the constraints. If all constraints are satisfied, then the solution is treated as correct by the system. Constraint violations indicate concepts or principles that the student might be lacking or misunderstanding and hence constitute high priority targets for instruction. In the simplest implementation of CBM, each constraint is associated with a single, canned feedback message which is presented to the student when that constraint is violated; more sophisticated instructional mechanisms are possible. A key advantage of CBM is that the implementation of an intelligent tutoring system does not require empirical studies of students' errors, and there is no need for explicit encoding of such errors in the form of buggy rules or collections of misconceptions. The relevant universe of errors is defined implicitly by specifying the constraints: The set of errors is the set of possible constraint violations. The psychological rationale for the constraint-based view of learning from error is developed in (Ohlsson, 1996a, 1996b; Ohlsson, 2011). CBM has been implemented in a number of systems that teach a wide variety of subject matters (Mitrovic, 2006; Mitrovic, Martin & Suraweera, 2007; Mitrovic, 2010).

SQL-Tutor is a constraint-based tutoring system that assists university-level students in acquiring the skill to create useful queries in SQL, a dominant database query language. Students find this skill hard to learn. To write SQL queries, they need to understand the relational data model and the concepts of the SQL language itself, and also be familiar with the particular database management system. Creating queries is an open-ended, ill-defined task: there is no algorithm for how to write good queries. The task of writing queries is learned through large amounts of practice. SQL-Tutor offers students practice opportunities and gives them feedback on their performance. Currently the system contains close to 300 problems defined in the context of 13 different

databases. Each problem is assigned a complexity level, which ranges from 1 (easy) to 9 (most complex).

When a student submits a solution it is sent to the student modeler for analysis. To check the correctness of the student's solution, SQL-Tutor compares it to the correct solution, using a knowledge base of 700 domain-specific constraints. The short-term student model consists of the list of relevant constraints and the list of violated/satisfied constraints. The student modeler maintains histories for all constraints that were relevant to the problems the student worked on while interacting with the system. If the student makes errors, i.e., if he or she violates one or more constraints, SQL-Tutor provides feedback.

SQL-Tutor offers six levels of feedback that differ with respect to the amount of detail. When a student submits a query for the first time, the system only informs him or her whether the solution is correct (*simple feedback*). If a student submits a second incorrect query, he or she is automatically advanced to the second and third feedback levels. The second level (*error flag*) highlights the part of the solution that is incorrect. The third level (*hint*) points out where the error is located, explains why it is an error and refers the student to the domain principle (constraint) that was violated. The automatic progression of feedback levels ends at the hint level. To obtain yet higher levels of feedback, the student must request them. For example, the student can ask for the hint messages for all violated constraints (*all errors*), a *partial solution* that gives the correct version of one part of the solution that the student got wrong, or the full, ideal solution (*complete solution*).

SQL-Tutor is a mature system. It has been used regularly in database courses at the University of Canterbury (and elsewhere) since 1998. A suite of three of CBM tutoring systems (including SQL-Tutor) that focus on topics in databases, is a successful commercial product (Mitrovic, Suraweera, Martin & Weerasinghe, 2004). They have been available on the Addison-Wesley's DatabasePlace[1] Web portal since 2003 (Mitrovic et al., 2006), and have been accessed by more than 10,000 students. SQL-Tutor has been evaluated in numerous quantitative studies and found to be effective, causing significant improvement of students' performance (i.e. difference between the post- and pre-test scores) even after short period of use (Mitrovic & Ohlsson, 1999; Mitrovic, 2003; Mitrovic, Martin & Mayo, 2002). The question investigated here is whether extending SQL-Tutor with a positive feedback capability improves students' learning over and above what is obtained with the standard, learning-from-error-only version of the system.

3.2. Implementing a Positive Feedback Capability in SQL-Tutor

Although constraint-based modeling was originally conceived as a technique for supporting teaching to errors, a constraint base can also be utilized to support positive feedback. We implemented an approach to positive feedback that addresses the issues of triggering condition and content of feedback messages within SQL-Tutor.

The triggering of positive feedback. The operation of the positive feedback capability is analogous to the way in which constraint-based systems handle negative feedback. Feedback is delivered in response to a submitted answer or solution. The first

[1] http://www.aw-bc.com/databaseplace/

of the six levels of feedback, simple feedback, is unchanged: The first time the student submits a solution for a particular problem, the system only informs the student whether the solution is correct. At this level, the tutoring message is thus the same as in the standard, learning-from-error-only version of SQL-Tutor. The main differences between the basic and the extended versions of SQL-Tutor are located in the higher levels of feedback.

A submitted solution usually exhibits some correct and some incorrect parts or aspects. Negative feedback is presented regarding incorrect aspects of a solution, even if other aspects of that solution are correct. Analogously, positive feedback does not require a completely correct solution. Instead, constraints that are satisfied by the student's solution can trigger positive feedback even if the solution violates other constraints.

It is not obvious which satisfied constraints are plausible targets for positive feedback. Most relevant constraints will be satisfied most of the time, and SQL-Tutor has a constraint base of over 700 constraints. To respond to a submitted solution with positive feedback pertaining to each and every constraint that is relevant and satisfied would overwhelm the student with unnecessary instructions. Instead, we developed rules for selecting solutions and constraints that are plausible targets for positive feedback. The selection depends not only on the submitted solution, but also on the student's knowledge (as captured by the long-term student model) and on the state of his or her interaction with the system.

Consistent with our main hypothesis, the triggering rules were designed to recognize opportunities for uncertainty reduction. The rules are designed to identify situations in which a student produces a solution that has some correct features, but he or she is hesitant or uncertain. The evidence for uncertainty takes several different forms: transitions from violation to satisfaction of a constraint, constraint satisfaction in response to a hint and constraint satisfaction in response to a difficulty.

(a) Transition from violation to satisfaction. In the basic case, some constraint C is satisfied for the first time. More precisely, the student has submitted one or more solutions $S_1, \ldots, S_{n-1}$, to a problem, and a constraint C was relevant in at least some of those solutions, but whenever it was relevant, it was violated. The student then submits a solution S_n in which C is satisfied. The occurrence of one or more violations prior to the current solution S_n is evidence that the student had not learned or understood the concept or principle expressed in C prior to submitting S_n. The fact that S_n satisfies C suggests that the student now has acquired that concept or principle, presumably on the basis of the negative feedback provided in response to the constraint violations in the prior solutions. The student's newly acquired knowledge has a high probability of being uncertain, and hence might benefit from confirmation through positive feedback.

There are several variations on this basic uncertainty reduction scheme. Students sometimes present a solution that satisfies some constraint C, only to submit one or more subsequent solutions that violate that same constraint. Either the constraint was satisfied accidentally or for the wrong reason, the student only understood how to satisfy the constraint in some special case, or the relevant knowledge was subject to interference, decay or retrieval failure. The inconsistency is evidence that the student is uncertain about the concept or principle expressed in C. If the student then submits a solution in which C is satisfied again, it is plausible that positive feedback might help consolidate his or her knowledge of C. The difference to the basic case is that in this scenario the current

solution does not satisfy C for the very first time, but after several solutions that violated C, which in turn followed one that satisfied C.

(b) Impasse-hint-constraint satisfaction (a.k.a. "drag and praise"). The uncertainty reduction principle also applies to situations in which the student is unable to produce a solution that satisfies some constraint C on his or her own. In this type of scenario, the student submits multiple solutions that violate C, receives negative feedback but nevertheless continues to submit solutions that violate C. One possible outcome is that the student moves to the next level of feedback by requesting a hint. If the solution submitted after receiving the hint satisfies C, then it is likely that the student just acquired the concept or principle expressed in C and he or she might benefit from positive feedback. If the student instead reaches an impasse, defined as a period of 5 minutes of inactivity, SQL-Tutor automatically takes the initiative and presents a hint. Once again, if the solution submitted after receiving the hint satisfies C, then C is a likely candidate for positive feedback.

(c) Overcoming difficulties. The uncertainty reduction principle can be combined with the notion of task difficulty to define a third type of scenario that is a plausible target for positive feedback: A student might be uncertain when he or she has just satisfied a difficult constraint[2]. One rule for presenting positive feedback is that whenever a student satisfies a constraint that is rated as difficult, then positive feedback that pertains to that constraint is presented. The difference to the basic uncertainty reduction case is that for difficult constraints, feedback is given when the constraint is satisfied for the first time, even if the solution was not preceded by solutions in which it was violated.

(d) Acknowledging task completion. A final triggering rule for positive feedback is that whenever a student submits a completely correct solution to a problem, his or her achievement is recognized with positive feedback message.

3.3. Content of Positive Feedback Messages

In the standard, learning-from-error-only version of SQL-Tutor, the content of negative feedback messages is closely tied to the violated constraints. A negative feedback message states the constraint, or the principle expressed in the constraint, in general form. In addition, such a message highlights how the student's solution violates that constraint (Zakharov, Mitrovic & Ohlsson, 2005).

The positive feedback messages follow an analogous format; see Table 2. The first part of each message is a content-free affirmative phrase ("Great work", "That's correct", etc.). The second part is a statement of the relevant constraint. The third part is a specification of how or why the student's step or solution satisfies the constraint. Examples 1 and 2 in Table 2 follow this format closely. In Example 1, "Well done!" is the affirmative phrase. The sentence "You now have all the tables needed in the FROM clause." rehearses the constraint that the student's solution satisfies. The first two sentences in Example 3 are analogous. The third sentence in Example 3 illustrates a message that pinpoints how the student's solution satisfies the relevant constraint: "var1" is a variable that will be bound to the relevant part of the student's solution. This elaboration provides the student with the reason why his or her solution is correct.

[2] The domain expert labeled 69 of SQL-Tutor's constraints as the difficult ones.

Example 3 in Table 2 illustrates the type of feedback given after the student solves a problem rated as very difficult (i.e. problems of complexity levels 7-9).

Place Table 2 here

A submitted solution can violate and satisfy multiple constraints, so there is a question of how to combine the instantiated templates that pertain to individual constraints into the complete feedback message that is shown to the student. If there are multiple violated constraints, SQL-Tutor selects one constraint on the basis of the pedagogical priority ordering ratings and generates negative feedback for that constraint. Likewise, if there are multiple satisfied constraints that fit the criteria for positive feedback, the system selects the first one and generates positive feedback for that constraint. The negative feedback capability was also operating, so if a submitted solution satisfies both types of criteria, then the system presents one positive and one negative feedback message. In those situations, the positive part of the message is always presented first. For example, Figure 1 illustrates a situation in which the student received some positive feedback about join conditions, and then negative feedback, on the *error flag* level, pointing the student to a mistake in the FROM clause.

Place Figure 1 here

In short, the extended version of SQL-Tutor is capable of delivering both positive and negative feedback. The new feature is that the constraint set, previously used to recognize students' errors, is also used to identify situations in which student are likely to be uncertain about the correctness of their solutions. As in the case of negative feedback, the constraints are used to determine the content of the positive feedback messages. The empirical question is whether this extension affects the students' learning gains.

4. Empirical Evaluation

An evaluation study was conducted at the University of Canterbury in Christchurch, New Zealand, with students enrolled in an introductory database course that lasted approximately one month, from May 9, 2007 to June 6, 2007. The participants logged on to SQL-Tutor for the first time during scheduled class time, but could use the system later at any time before the end of the course. The participants were randomly assigned to either the control group or the experimental group. The control group students interacted with the standard, errors-only version of SQL-Tutor and hence only received negative feedback, while the students in the experimental group interacted with the extended version and hence received both negative and positive feedback. Out of 79 students

enrolled in the course, 55 logged on to SQL-Tutor at least once, and 51 completed the on-line pre-test, 47 male and 4 female. We eliminated the logs of 10 students who used the system for less than 10 minutes. Of the remaining 41 students, 18 were assigned to the experimental group, all males, and 23 to the control group, 21 males and 2 females.

All student actions were recorded in the student-computer interaction logs. Table 3 shows the means and standard deviations for some descriptive variables. The maximum score on the pre-test was 4. There was no significant difference between the mean scores on the pre-test for the two groups. The two groups attempted approximately the same number of problems. The control group submitted a larger number of proposed solutions than the experimental group, but the difference was not significant. The control group received more negative feedback messages than the experimental group, but the difference was not significant. The students in the experimental group saw an average of 22 positive feedback messages. The total number of feedback messages seen was 222 for the control group and 204 for the experimental group. The control group thus receive slightly more instruction than the experimental group, but the difference was not significant.

Table 3 in here

Table 4 shows the means and standard deviations for the outcome variables. As shown in the first row of Table 4, the number of constraints learned was approximately the same for the two groups. The criterion that a constraint was learned consisted of two parts: First, that the constraint was not known at the outset, as indicated by the fact that it was violated on at least 3 out of the first 5 occasions on which it was relevant. Second, that the constraint was learned before the end of the study, as indicated by it being satisfied in at least 3 out of the last 5 occasions on which it was relevant. On this criterion, the students acquired the concepts and principles expressed in approximately 10 constraints in both groups. Similarly, the two groups did not differ significantly with respect to the number of practice problems that were solved correctly. In short, the two groups acquired approximately the same amount of knowledge.

Table 4 in here

The picture is radically different with respect to the time to acquisition. The third row of Table 4 shows the total amount of time spent interacting with the system. Recall that the study was conducted in a class setting. Although we assigned students to either the control or the experimental group, we did not have control over how much time the students spent interacting with the tutoring system. As it turned out, the experimental

group spent half as much time as the control group, 92.3 minutes as compared to 193.8 minutes. Mastery was reached in 60% of the time; this result is shown in terms of time per correctly solved problem and time per attempted problem, displayed in rows four and five in Table 4. The result is the same regardless of which outcome variable is used: The time to mastery for the experimental group was approximately half the corresponding time for the control group.

The last row in Table 4 shows the average score on the subsequent examination for the course. The advantage of the experimental group did not transfer from the tutoring system to the examination. There is no significant difference in the examination scores. The reason for this outcome is that the examination score, unlike the other outcome variables, is highly correlated with the pre-test score. That is, the examination problems primarily tested the students' prior knowledge, as opposed to what they learned from the instruction, while the opposite is true for the constraint-based outcome measure.

5. Conclusions

Our result supports the conclusion that there is pedagogical power in the simple act of confirming to a student that a problem-solving step just taken was in fact appropriate, correct, or useful. Students who received both positive and negative feedback mastered a certain amount of new knowledge, measured in constraint learned, in half the time required by students who received negative feedback only. Such big decreases in the total interaction time and the time to mastery (measured by the time needed per solved problem) are huge, pedagogically significant outcomes. The results suggest that intelligent tutoring systems that teach primarily by addressing errors and misconceptions might become yet more helpful if extended with a positive feedback capability. It is important to keep in mind that we are not faced with an either-or situation. Our results should not be interpreted as saying that positive feedback is *more effective than* negative feedback, or that tutoring systems should deliver positive *instead of* negative feedback. The result suggests that a tutoring system that teaches to student errors can be improved by adding a capability to teach to their successes as well.

The particular way in which the positive feedback capability was implemented in SQL-Tutor is not the only technique for implementing this type of instruction; it is a single point in a large design space. Given the hypothesis that positive feedback helps by reducing student uncertainty, the central design issue is how a tutoring system can recognize that a student is uncertain. In SQL-Tutor, the constraint base originally implemented to support negative feedback was used for this purpose. It is possible that other types of knowledge representations can serve the same function as well or better.

For example, we envision a rule-based system that uses model-tracing in the analogous way as the constraints were used in SQL-Tutor: Such a system would contain a rule base that encodes the correct strategies for the domain, and matches student steps to correct (rather than buggy) rules. If a student step matches a correct rule for the first time, it is a plausible inference that it is associated with some level of uncertainty and so signals an opportunity to instruct with positive feedback. A model-tracing system that functions in this way would not need a knowledge base of buggy rules, and hence could dispense with empirical studies of common student errors, one of the great burdens on those who undertake to build model-tracing systems. An intelligent tutoring system that

uses constraints to identify constraint violations and hence opportunities for negative feedback while basing positive feedback on model-tracing with a knowledge base of correct rules might be powerful.

An alternative technique was developed by Fossati et al. (2009) for use in iList, an intelligent tutoring system that teaches elementary data structures (lists, stacks, and trees). iList represents the space of possible moves – the problem space – in a tree structure. Rather than representing the entire space, iList compiles the portion of the space that students actually visit by combining the log files of students who worked with a preliminary version of the system. There is a separate tree structure for each practice problem. The information in the log files is used to calculate the probability that a student who has reached node N in the tree for practice problem X will ultimately complete problem X successfully. This probability is estimated as the proportion of students in the past who passed through state N and completed the problem correctly. (This is a simplified description. The actual calculations used by iList are available in Fossati et al., 2009.) When a student moves from state N to state N' in the problem space, iList can consult these estimates to decide whether the step taken was productive. If the student moves to a node in which the probability of completing the problem is as high as, or higher than, in the previous node, the student is probably on the right track. If he or she also remained in node N for an above-average amount of time, indicating hesitation, then the situation calls for positive feedback to encourage the step taken. On the other hand, if the student moves to a node that has a lower probability of being on the path to the correct solution, then the tutor should either remain silent or issue a warning or negative feedback. This approach presupposes a pilot study to generate a set of log files from which the relevant parts of the problem spaces can be computed. It also requires that the procedure is repeated when new practice problems are added to the system.

The above techniques for identifying opportunities to deliver positive feedback rely on some form of knowledge tracing over a period of time. The problem can also be approached via local empirical measures. An obvious indicator of hesitation is response time. If a student spends significantly longer time making up his or her mind what to do next in one situation than in general, then it is reasonable to infer that he or she was hesitant, so if the step he or she finally did take was in fact correct, positive feedback might be useful. A more direct approach is to ask students for confidence judgments in connection with each step, and define a level of confidence that indicates the positive feedback should be delivered if the step turns out to be correct. This latter technique is likely to be more or less disruptive, depending on the nature of the domain and the types of actions available in the task interface. Psychophysiological measures like skin conductance might be worth exploring, while machine vision powerful enough to recognize furrowed brows is probably some distance into the future.

Given some machinery for identifying situations in which positive feedback might be useful to the student, a key design problem is what to do when a student supplies a problem solving step or a solution that contains some erroneous and some correct features. If both positive and negative feedback are relevant, how does the tutoring system choose between them? The simplest solution is to have a fixed priority ordering: If negative feedback is indicated, always present it. In SQL-Tutor, the decision rule was that if both positive and negative feedback were indicated, the system presented one positive and one negative message, with the positive message presented first. Other

decision procedures can be envisioned. For example, messages might be rank ordered with respect to pedagogical importance and presented in order of importance. An advanced decision rule would consult the actual content of the errors and the successes and compute a decision on line, but this type of mechanism is as yet in the future.

References

Anderson, J. R., Corbett, A. T., Koedinger, K. R., & Pelletier, R. (1995). Cognitive tutors: Lessons learned. *Journal of the Learning Sciences, 4*, 167-207.

Boyer, K. E., Phillips, R., Wallis, M., Vouk, M., & Lester, J. (2008). Balancing cognitive and motivational scaffolding in tutorial dialogue. In B. P. Woolf, E. Aïmeur, R. Nkambou, & S. Lajoie (Eds.), *Proceedings of the 9th International Conference on Intelligent Tutoring Systems* (pp. 239-249). New York: Springer-Verlag.

Brown, J. S., & Burton, R. R. (1978). Diagnostic models for procedural bugs in basic mathematical skills. *Cognitive Science, 2*, 155-192.

Cade, W. L., Copeland, J. L., Person, N. K., & D'Mello, S. K. (2008). Dialogue modes in expert tutoring. In B. P. Woolf, E. Aïmeur, R. Nkambou & S. Lajoie (Eds.), *Proceedings of the 9th International Conference on Intelligent Tutoring Systems* (pp. 470-479). New York: Springer-Verlag.

Chi, M. T. H., Siler, S. A., Jeong, H., Yamauchi, T., & Hausmann, R. G. (2001). Learning from human tutoring. *Cognitive Science, 25,* 471-533.

Di Eugenio, B., Fossati, D., Ohlsson, S., & Cosejo, D. G. (2009). Towards explaining effective tutorial dialogues In N.A. Taatgen & H. van Rijn (Eds.), *Proceedings of the 31st Annual Conference of the Cognitive Science Society* (pp. 1430-1435). Austin, TX: Cognitive Science Society.

Fossati, D., Di Eugenio, B., Brown, C. W., Ohlsson, S., Cosejo, D. G., & Chen, L. (2009). Supporting computer science curriculum: Exploring and learning linked lists with iList. *IEEE Transactions on Learning Technologies, 2*, 107-120.

Koedinger, K. R., Anderson, J. R., Hadley, W. H. & M. A. Mark (1997). Intelligent tutoring goes to school in the big city'. *International Journal of Artificial Intelligence in Education, 8*, 30-43.

Mitrovic, A. (2003). An intelligent SQL tutor on the Web. *International Journal of Artificial Intelligence in Education, 13,* 173-197.

Mitrovic, A. (2006). Large-scale deployment of three intelligent web-based database tutors. *Journal of Computing and Information Technology, 14*, 275-281.

Mitrovic, A. (2010). Modeling domains and students with constraint-based modeling. In: R. Nkambou, J. Bourdeau & R. Mizoguchi (Eds.), *Advances in Intelligent Tutoring Systems*, Berlin Heidelberg: Springer, pp. 63-80.

Mitrovic, A., Martin, B., & Mayo, M. (2002). Using evaluation to shape ITS design: Results and experiences with SQL-Tutor. *International Journal of User Modeling and User-Adapted Interaction*, *12*, 243-279.

Mitrovic, A., Martin, B., & Suraweera, P. (2007). Intelligent tutors for all: Constraint-based modeling methodology, systems and authoring. *IEEE Intelligent Systems*, *22*, 38-45.

Mitrovic, A. & Ohlsson, S. (1999). Evaluation of a constraint-based tutor for a database language. *International Journal of Artificial Intelligence in Education*, *10*, 238-256.

Mitrovic, A., & Ohlsson, S. (2006). Constraint-based knowledge representation for individualized instruction. *Computer Science and Information Systems*, 3, 1-22.

Mitrovic, A., Suraweera, P., Martin, B., & Weerasinghe, A. (2004). DB-suite: Experiences with three intelligent, web-based database tutors. *Journal of Interactive Learning Research*, *15*, 409-432.

Ohlsson, S. (1992). Constraint-based student modeling. *Journal of Artificial Intelligence and Education*, *3*, 429-447.

Ohlsson, S. (1996a). Learning from performance errors. *Psychological Review*, *103*, 241-262.

Ohlsson, S. (1996b). Learning from error and the design of task environments. *International Journal of Educational Research*, *25*, 419-448.

Ohlsson, S. (2008). Computational models of skill acquisition. In R. Sun (Ed.), *The Cambridge handbook of computational psychology* (pp. 359-395). Cambridge, UK:

Ohlsson, S. (2011). *Deep learning: How the mind overrides experience*. New York: Cambridge University Press.

Ohlsson, S., & Di Eugenio, B., Chow, B., Fossati, D., Lu, X., & Kershaw, T. (2007). Beyond the code-and-count analysis of tutoring dialogues. In R. Luckin, K. Koedinger & J. Greer (Eds.), *Artificial intelligence in education: Building technology rich learning contexts that work* (pp. 349-356). Amsterdam, The Netherlands: IOS Press.

Ohlsson, S., & Mitrovic, A. (2007). Fidelity and efficiency of knowledge representations for intelligent tutoring systems. *Technology, Instruction, Computers and Learning*, *5*, 101-132.

Sleeman, D., Kelly, A. E., Martinak, R., Ward, R. D. & J. L. Moore (1989). Studies of diagnosis and remediation with high school algebra students. *Cognitive Science, 13*, 551-568.

Spohrer, J. C., Soloway, E., & Pope, E. (1985). A goal/plan analysis of buggy Pascal programs. *Human-Computer Interaction, 1*, 163-205.

Zakharov, K., Mitrovic, A. & S. Ohlsson (2005). Feedback micro-engineering in EER-Tutor. In C.-K. Looi, G. McCalla, B. Bredeweg, & J. Breuker (Eds.), *Proceedings of the 12th International Conference on Artificial Intelligence in Education*, IOS Press, (pp. 718-725). Amsterdam, The Netherlands.

List of Table Captions

Table 1. Four positive feedback episodes excerpted from Java programming tutoring sessions.

Table 2. Three examples of positive feedback messages from the extended version of SQL.

Table 3. Means and standard deviations for six descriptive variables.

Table 4. Means, standard deviations and differences in six outcome variables between two experimental groups.

Table 1. Four positive feedback episodes excerpted from Java programming tutoring sessions (provided by Kristy Elizabeth Boyer).

No	Type of Message	Speaker	Utterance
1	Confirmation of correctness	Student	So I would use a *for* loop instead of writing out the digits five times.
		Tutor	Exactly.
2	Prompt and student uncertainty	Tutor	Now we want to take each character from the string and put the right integer into the string.
		Student	With a *for* loop?
		Tutor	Yes
		Student	Under *extractDigits*?
		Tutor	Yep
3	Prompt and student uncertainty	Tutor	The only thing is that you are trying to store *chars* in a *int* array; how can you fix that?
		Student	Can I parse a character to an *int*?
		Tutor	Yup.
		Student	Typing *table* is acceptable here?
		Tutor	That's what I would do.
4	Conceptual elaboration	Tutor	There are other examples that the top of the class.
		Student	Like that?
		Tutor	Yup!
		Tutor	You don't have to do the *mod* stuff for the *cdigit* until the next method.

Table 2. Three examples of positive feedback messages from the extended version of SQL-Tutor.

No.	Feedback message
1	"Well done! You now have all the tables needed in the FROM clause."
2	"That's correct! All the names used as attributes names in the WHERE clause must be attribute names from the tables named in the FROM clause. " 'var1 " is a valid attribute name in the FROM clause."
3	"Fantastic, you solved a very difficult problem in very few attempts!"

Table 3. Means and standard deviations for six descriptive variables.

	Group		
Variable	Control	Experimental	*t*-test
Pre-test score	1.7 (0.9)	2.1 (1.1)	*ns*
No. of problems attempted	28 (25)	26 (15)	*ns*
No. of solutions submitted	119 (99)	98 (66)	*ns*
Negative messages received	222 (169)	182 (127)	*ns*
Positive messages received	-	22 (12)	N/A
All feedback messages	222 (169)	204 (138)	*ns*

Table 4. Means, standard deviations, and differences in six outcome variables between two experimental groups.

Outcome variable	Group		*t*-test
	Control	Experimental	
Constraints learned	10 (6.1)	9.3 (6.8)	ns
Problems solved	25 (24)	22 (15)	ns
Total time on system (min)	194 (198.7)	92.3 (44.7)	$p < 0.02$
Time per solved problem (min)	9.8 (7.9)	5.8 (4.8)	$p < 0.03$
Time per attempted problem (min)	7.5 (4.5)	4.1 (2.0)	$p < 0.01$
Exam score (% of max. score)	57 (26.5)	59.3 (24.3)	ns

Figure 1

Click here to download high resolution image

Acknowledgments

The preparation of this article was supported, in part, by Award No. N00014-07-1-0040 from the Office of Naval Research (ONR), US Navy, to the second author, and by a strategic grant from the University of Canterbury to the first author. No endorsement should be inferred. We thanks Kristy Elizabeth Boyer for providing excerpts from her Java tutoring corpus. We wish to thank our colleagues in the Intelligent Computer Tutoring Group (ICTG) at the University of Canterbury as well as the Natural Language Tutoring Group at the University of Illinois at Chicago for stimulating discussions regarding tutoring.

***Highlights**

- We discuss the role of positive feedback in intelligent tutoring systems
- We present an implementation of positive feedback in a constraint-based tutor
- We compared the system with negative feedback to one with both types of feedback
- The results show that providing positive feedback cuts the time to mastery in half